HIP Mobile Client

Perttu Myry

Master's Thesis
May 2013

Degree Programme in Information Technology
Technology and transportation

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

DESCRIPTION

Author(s)	Type of publication	Date
MYRY, Perttu	Master's Thesis	28.05.2013
	Pages 55	Language English
		Permission for web publication (X)

Title
HIP Mobile Client

Degree Programme
Information Technology

Tutor(s)
MANNINEN, Pasi

Assigned by
Humap Software Ltd

Abstract
HIP is an acronym that stands for Humap Inspiration Platform. It is a web-based service developed and provided by Humap software Ltd. It can be used to build various products, services and concepts for numerous purposes.

In this thesis the process of creating a mobile client for this platform is discussed in detail. HIP has been developed since 2008 and its predecessor Humap Tool since 2000. Therefore, there are hundreds of features implemented in the platform. It was vital to research what the most used features in the platform are and which one of them would be most needed and most useful to be implemented in the mobile client. Furthermore, the concepts that can be built using the platform can vary a lot and only certain core level features are used in most cases.

Additionally, technology research was conducted in which the concurrent mobile development platforms and frameworks were investigated and evaluated. By comparing the most used mobile development tools with each other it was possible to pick the best combination to be used in order to implement HIP mobile client as easily and quickly as possible.

The outcome was a simple application which is generic enough to fit into most concepts that can be created using the platform.

Keywords
Mobile, jQuery Mobile, PhoneGap

Miscellaneous

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖN
KUVAILULEHTI

Tekijä(t) MYRY, Perttu	Julkaisun laji Opinnäytetyö	Päivämäärä 28.05.2013
	Sivumäärä 55	Julkaisun kieli Englanti
		Verkkojulkaisulupa myönnetty (X)

Työn nimi
HIP Mobile Client

Koulutusohjelma
Degree Programme in Information Technology

Työn ohjaaja(t)
MANNINEN, Pasi

Toimeksiantaja(t)
Humap Software Oy

Tiivistelmä
HIP on lyhenne sanoista Humap Inspiration Platform. Se on Humap Software Oy:n kehittämä ja tarjoama web-pohjainen alusta, jonka avulla voidaan rakentaa varsin monimuotoisia tuotteita, palveluita ja erilaisia konsepteja.

Tässä opinnäytetyössä käydään läpi prosessi kuinka tälle alustalle luotiin mobiilisovellus. HIP alustaa on kehitetty vuodesta 2008 ja sen edeltäjää Humap Tool palvelua vuodesta 2000 alkaen. Tämän vuoksi alustaan on vuosien saatossa kertynyt satoja ominaisuuksia. Onnistuneen lopputuloksen kannalta oli äärimmäisen tärkeää tutkia, mitkä olivat alustan käytetyimmät ominaisuudet ja mitkä niistä olisivat kaikista tärkeimpiä ja hyödyllisimpiä toteuttaa mobiilisovellukseen. Tämän lisäksi eri konseptit, joita alustan avulla voidaan kehittää, voivat poiketa toisistaan merkittävästi. Vain tietyt ydinominaisuudet ovat käytössä lähes kaikissa alustalla tehdyissä toteutuksissa, mikä osaltaan ohjasi valitsemaan tietyt ominaisuudet mobiilisovellukseen.

Tämän lisäksi myös nykyaikaisia mobiilisovellusten kehitystyökaluja tutkittiin ja arvioitiin. Vertailemalla eniten käytettyjä kehitystyökaluja keskenään oli mahdollista löytää juuri oikeat työkalut, joiden avulla mobiilisovellus voitiin toteuttaa mahdollisimman helposti ja nopeasti. HIP alustassa on käytössä jQuery-kirjasto, joten se on tuotekehitystiimille entuudestaan tuttu ja siksi jQuery Mobile oli helppo ja luonteva valinta sovelluksen pohjaksi. PhoneGap sen sijaan tarjosi helpon tavan tehdä sovelluksesta natiivi applikaatio usealla eri mobiililaitteelle käyttämällä yksinkertaista pilvipalvelua.

Lopputuloksena oli yksinkertainen ja toimiva sovellus, joka on riittävän geneerinen sopiakseen useimpiin konsepteihin, joita alustan avulla voidaan toteuttaa.

Avainsanat (asiasanat)
Mobiili, jQuery Mobile, PhoneGap

Muut tiedot

Contents

Figures

Terminology

API (Application Programming Interface) provides developers a set of instructions and functions to do certain actions in software.

CRM (Customer Relationship Management) is a model for managing customers which also includes software to manage this information.

CSS (Cascading Style Sheets) is a style sheet language used for decorating and defining the layout of web pages.

ERP (Enterprise Resource Planning) systems are used to manage both organizations' internal and external information in one single system.

HRM (Human Resource Management) means management of organizations' human resources.

HTML (HyperText Markup Language) is a markup language used for creating web pages.

IDE (Integrated development environment) is an application that provides environment for software development.

JS (JavaScript) is a programming language used in mainly web pages for creating dynamic functionality and content.

Lean is a production practice in which the "core idea is to maximize customer value while minimizing waste". In other words "lean means creating more value for customers with fewer resources". (What Is Lean 2009.)

OS (Operating System) "is a collection of software that manages computer hardware resources and provides common services for computer programs" (Operating system, 2013).

PMS (Project Management Software) is software used for managing projects.

Product Backlog is a Scrum term referring to an ordered list of everything that might be needed in the product.

Scrum is an agile software development method.

UI is an abbreviation of user interface.

We and **us** in this thesis refers to employees of Humap Software Ltd.

1 Introduction

1.1 Reasons for Choosing Mobile Related Topic

When I started to consider a topic for this thesis it was easy for both me and the host company Humap Software Ltd to state the reasons why this thesis could be related to mobile technology.

Personally my interest towards mobile development was not sky-high before I got a smartphone of my own. That really changed the game. Once I had such a device in my pocket all the time I just had to get into mobile development. My background is in web development so extending my skillset with mobile expertise was a natural path to follow. I was mostly familiar with the programming languages and technologies that were related to the mobile client. However, the mobile world with all its different devices, operating systems, ways of using applications and context in which applications are used meant quite a few challenges for me. Anyway, I was eager to encounter each and every one of them.

From business point of view nobody should overlook the sheer amount of mobile devices and users which have been increasing dramatically during last couple years. In case you do not have anything to offer to mobile users you are closing the door for ever increasing amount of users. Companies need to carefully consider what to offer for mobile users and how, because mobile world differs significantly from traditional desktop computer usage and it is easy to burn a lot of time to mobile implementations without offering real added value to mobile users. Then again, a successful mobile strategy means maximum potential in terms of amount of users and availability of your services.

Humap is a small company so an agile and lean strategy was needed. In this thesis I will show how it is possible to create mobile client that works in just about every mobile device, will offer the right features for mobile users, and still you do not have to invest huge amount of resources in it.

1.2 Host Company

In order to understand the ideology of the host company Humap Software Ltd the history of the parent company Humap Ltd is presented, how and why it was founded in the first place.

The story of Humap starts at the end of summer 1998. Three friends, Olli-Pekka Juoperi, Ilkka Mäkitalo and Vesa Purokuru with similar backgrounds in teacher studies, were giving a team based training session somewhere in Finland. When they were driving back home after the training they stopped at a gas station and started wondering whether they would consider themselves as a team. They ended up with the conclusion that they were not yet a team; however, there was a clear interest towards working together. They first started collaborating so that they each had their own legal business names, but rather soon they decided to start working under the same company and Humap Ltd was founded in April 1999. (Havimäki 2012.)

The main purpose of Humap was to develop Finnish education system. In addition, management level training and consultation was a clear goal from the very beginning of the company. The founders did not pursue towards getting rich, they were more like dreamers who wanted to live an interesting life, harvest new experiences and create the best possible place to work. (Havimäki 2012.)

Web technology has been a part of Humap since the very beginning of the company. In a way, Humap has been developing social media implementations before the concept became popular. The technical solutions were tightly coupled with training and consultation and these two sides of the company, consultation and technology are still an essential part of Humap today. (Havimäki 2012.)

For roughly a decade consultation and technology walked hand in hand merged into one single company. In 2008 things changed when the technology aspect of the company was separated to its own subsidiary Humap Software Ltd. This meant more clarity in terms of accounting and financial tracking; however, the two companies still work closely together providing both technological solutions and consultation.

Humap Software is not just any kind of software development company. The company's history and close collaboration with Humap can be seen in daily work. The company's main goal is to change the way people think and technology is a tool to achieve this goal and assist people in their everyday tasks. This all is summarized quite well in this short introduction at Humap's website.

> *Work is becoming more complex. Success requires high-quality collaboration. Technology creates prerequisites, but it is not however enough. We need to change the way we think and we need new efficient ways of working. We help to renew, excite and succeed. (Humap – New Ways of Working 2012.)*

1.3 Mobile Technology

It is highly likely that you are carrying a mobile phone with you The author of this thesis makes this assumption because in 2011 there were 5972 million mobile-cellular subscriptions registered worldwide (Key ICT indicators for developed and developing countries and the world 2013). One could also visualize these numbers as if 85.7 % of the world's population has been carrying a mobile phone in 2011. Of course, this is not true as these numbers do not take into account that some people have various mobile devices and not all registered mobile-cellular subscriptions are mobile phones; however, it gives an idea how popular mobile devices have become. It has been predicted that during 2013 more people will use their mobile phones than PCs to get online and by year 2015 there will be one mobile device for every person on earth (Reasons Mobile Matters 2011).

Unlike TV or computer, mobile phone is a device that is with the user almost all the time. And what is intriguing from developer's point of view is that mobile technology has developed drastically over the past years. Modern smartphones have powerful hardware and they can run rather complex software.

But what is actually a mobile device? This is actually an excellent question. Currently there are various devices on the market which could be considered as mobile devices such as music players, mobile phones, tablet devices and small laptops such as netbooks and ultrabooks (see Figure 1). In order to narrow the list the same list of mo-

bile device features is utilized in this thesis as Firtman (2012, 4) lists in his book Programming the Mobile Web on page four:

- It is portable.
- It is personal.
- It is with you almost all the time.
- It is easy and fast to use.
- It has some kind of network connection.

FIGURE 1. Some mobile devices including iPhone (iOS), HTC Sensation (Android), iPad (iOS), Nokia Lumia 820 (Windows Phone) and BlackBerry (BlackBerry OS)

By reflecting few examples on this list it can be said that laptops, regardless of their size, are not mobile devices as it is likely that they are not with you all the time and they are not easy and fast to use at least in comparison to a mobile phone. On the other hand, music players rarely have a network connection so that rules out most of these devices. Then again, a device like iPod Touch can be considered as a mobile device as it fits in to all of the features listed above.

Tablets are somewhat hard devices to categorize. They fit quite well into all of the mentioned features. However, whether they are easy and fast to use needs a short revision. In the book Programming the Mobile Web the following statement was made about easy usage.

> *A mobile device needs to be easy and quick to use. I don't want to wait two minutes for Windows to start; I don't want to sit down. If I'm walking downtown, I want to be able to find out when the next train will be departing without having to stop. (Firtman 2012, 5.)*

Tablet devices are somewhat on the edge when it comes to what this statement requires from mobile devices. Tablets are rather big devices, both hands are needed to operate them and it is not an easy task to handle a tablet device without stopping or sitting down. Nevertheless, tablets can be considered as mobile devices. For mobile developers this means though that different kind of use cases should be carefully thought through. Tablets have somewhat different kind of properties than traditional mobile devices and they are probably used in different kind of situations. Whereas someone might check out something like weather, bus schedules or short emails using phone while being on the move, a tablet would more likely be used on a bus or train or while sitting at a lobby of a hotel waiting for a colleague to arrive.

2 Introduction to HIP

2.1 Introduction

HIP is an acronym of Humap Inspiration Platform. It is web-based service which enables new ways of working in various kinds of networks. In order to really understand what the platform is it is necessary to dive back in history and go through the early days of Humap and what kind of path has led to creation of HIP.

2.2 History

As stated earlier, web technology has been a part of Humap since the very beginning of the company. In the early days of the company this meant implementing customer specific services that were always unique. Humap soon realized though that this business model had its own problems. Customers rarely had enough money to invest in developing their own product; therefore further development of the products did not work out too well.

Because of this issue people Humap started to wonder whether there could be another kind of solution in which they would still be able to produce highly customized services implementing them cost efficiently. Humap collected the best ideas from all the customer implementations done so far and this is how HIP's predecessor Humap Tool was emerged in 2000.

Even though modern social media surfaced back in 1990s and the first social media site Geocities was created in 1994, social media, or "SoMe" as it is often called, did not get major attention before services like wikis, blogs, Facebook, Twitter and Youtube were created (Social media 2013). For example, Facebook was created back in 2004 and it took some time for it to reach global success. This is why it is fair to say that Humap Tool was somewhat ahead of its time as it implemented social media principles before they were accepted and embraced by large audience.

Humap Tool (see Figure 2) is Wiki-like service the strengths of which are its simplicity and collaboration features. Humap Tool consists of a predefined set of tools which can be enabled or disabled for different kind of customer needs. Some of the most used tools in Humap Tool are customer relationship management tool, material bank, web surveys and chameleon which is a tool used for various purposes such as project management. In addition, there is a feature which can be used to create users' own tools based on the available building blocks such as editable text blocks, tag lists and comment areas. Humap Tool also has a layout editor which can be used to customize the layout to fit an organization's look and feel.

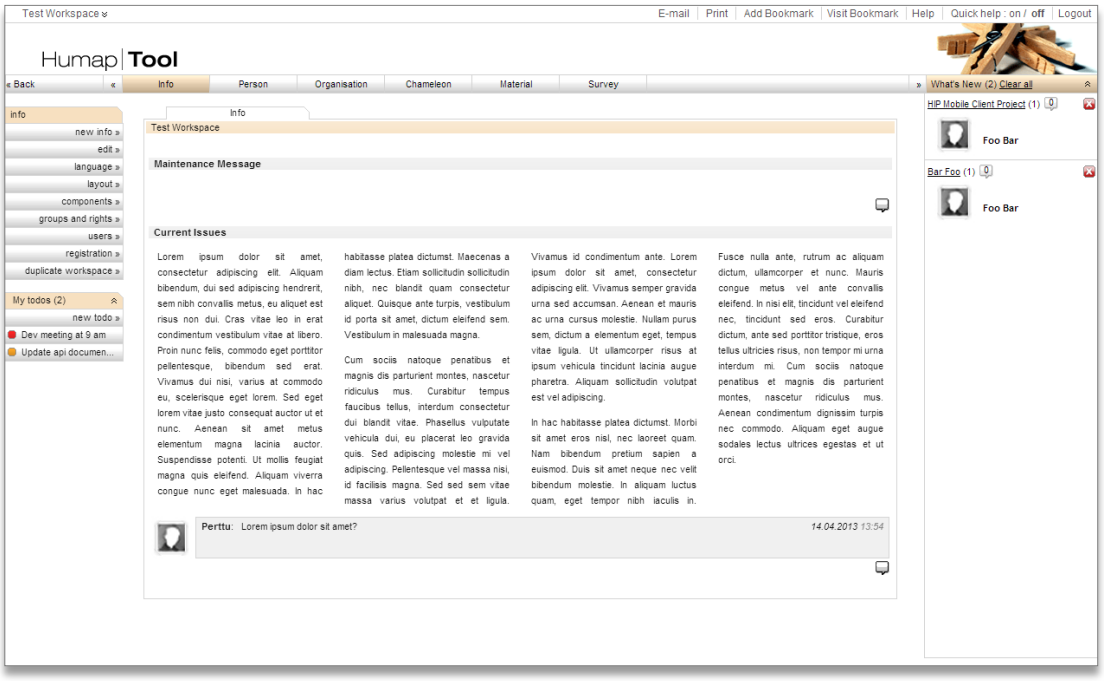

FIGURE 2. Humap Tool example workspace

The problem with Humap Tool was though that even if it is configurable and scalable, it was not really unique for each customer and there are limitations in terms of what can be done with it. In case some customer wanted a feature that was not already part of Humap Tool it required rather time taking product core level changes which were usually expensive to implement. In addition, throughout its lifecycle the product's architecture was not properly planned from the beginning and the codebase ended up being quite a puzzle to solve. Maintenance started to cause additional headache and developing new features became an increasingly complex operation due to tight coupling between different features. Because of these reasons there was a desire to create a completely customizable platform which could be used to build quickly and cost efficiently just about anything that a customer wants. Eight years after Humap Tool development was started these ideas went from brainstorming to implementation and on 10th of April 2008 the first code commits were made to HIP version control repository.

2.3 Ideology of the Platform

When the development of the platform at Humap was started there was a distinct goal to make sure that the existing Humap Tool customers would not be lost. Thus

HIP was integrated also as a part of Humap Tool although HIP can be and is primarily used also as a stand-alone platform. This way new customer cases could be implemented using just HIP and yet offer the platform's features to existing Humap Tool users as well. The following quote from Humap's website summarizes what HIP is all about.

> *HIP – Humap Inspiration Platform keeps up the tradition of Humap Tool. Our Humap Tool clients can continue using Humap Tool and combine it with new functions or replace old functions with new solutions. Our new clients can join us in building inspiring and fun web solutions solely on the HIP platform. HIP is a flexible innovation and inspiration platform for specialist work that is based on Enterprise 2.0 thinking. We can produce a solution that suits your needs quickly, cost effectively and efficiently by using flexible widget technology. (Humap Inspiration Platform – HIP 2012.)*

The statement "flexible and innovative web environment" is quite abstract; however, it really states quite well how dynamic and multifunctional the platform is. Many features have been built to the platform itself, the actual customer implementations; however, can utilize a very random subset of those features.

It makes no sense to list all its different use cases and give a full feature list, but for sake of clarity, here are few examples how HIP is currently being used. In the current implementations HIP based solutions have been used as intranet, extranet and public implementations with and without HIP's user and user group management system. HIP has been used, for example, to create CRM, PMS, HRM and ERP systems, material management systems, online surveys, value discussions, online communities, various kinds of collaboration solutions in various sized teams and organizations. These implementations can be also split into different categories which are visualized in Figure 3.

FIGURE 3. Different ways to utilize HIP

The acronym HIP actually hides more to itself than what might initially be perceived:

- "H" refers to Humap as a company, but also to humanity and to how Humap tries to find out what kind of issues and needs people face in their daily work and develop the best possible solutions to meet those needs.
- "I" is about inspiring people. Humap always tries to find solutions which inspire people and therefore help to create ideas, innovation and enthusiasm in to the organization.
- "P" comes from the platform which is built upon these humane principles. The platform is built so that it is possible to create inspiring solutions quickly and cost-effectively. P also means prototyping which is one of the centric working methods at Humap. Via doing quick tests and acquiring early experiences it is easier to develop the solution into such direction that it will suit users' needs as well as possible.

(HIP Platform - Ainutlaatuinen alusta innovaatioiden jalostamiseen 2013.)

2.4 Current State

HIP has been actively developed for over five years. With Humap Tool it was considered to rewrite the codebase after around seven years of development. HIP is nowhere near that state. This is possible because of good architecture and constant refactoring. Due to its modular structure it has been possible to add features to the platform without making it too complex to maintain or furthermore develop in the future.

Due to flexible template and widget structure the former problems faced with Humap Tool are now tackled in HIP. Customers are no longer forced to use certain kind of navigation, set of tools or certain structure, but instead they can tell what they need and it is possible to craft them exactly the kind of environment which fits their needs.

Furthermore, Humap Tool utilizes latin-1 character encoding and those past experiences taught the company to use UTF-8 in HIP which enabled support of wide variety of languages. Because of this it was possible to build a good localization support in HIP and customer implementations have been done in Finnish, British English, US English, Dutch, German, Russian, Latvian and Estonian languages. In addition, it was tested that the platform is technically ready for languages such as Arabic or Chinese in case Humap will expand into those market areas.

2.5 Concepts

HIP has many technical details and terms that will not be introduced in this thesis. The following concepts and terms though are so centric to HIP that it is vital to understand them so that one can understand some crucial parts of the mobile client.

2.5.1 Hierarchy of Concepts

The concepts will be introduced from the largest to the smallest. To clarify the whole idea of HIP the hierarchy of these concepts is visualized in Figure 4.

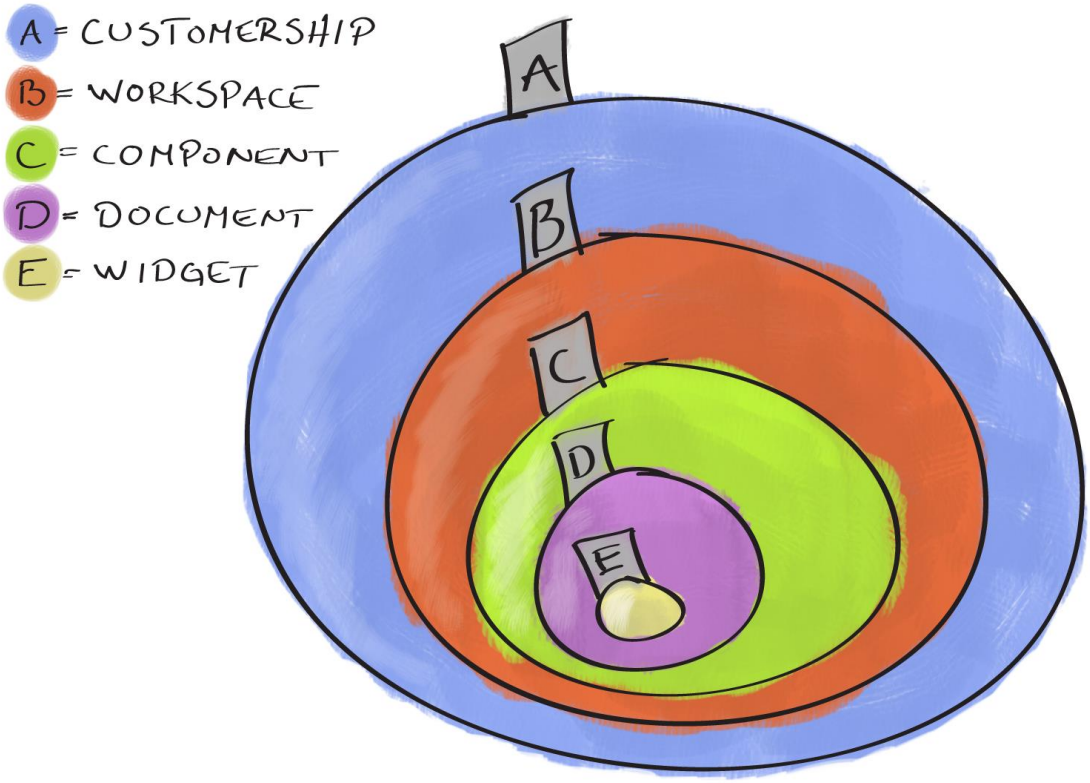

FIGURE 4. HIP concept and terminology visualized according to their hierarchy

2.5.2 Customership

On a single HIP server all information is categorized into customerships. This means that one customer has their own sandbox of information and they have no knowledge of other customers and cannot access their information in any case.

2.5.3 Workspace

A single customer can separate their own information into workspaces. Compared to an operating system, one could visualize workspace as a folder. The purpose of workspace can vary between different implementations. For example, if a customer organizes courses and they want to share course materials for course participants, they can put up a new workspace for the course, invite users to register into that workspace and have that piece of information separated from their own intranet. Alternatively if a company has multiple departments, they can use workspaces to separate them as workspaces so that each department has their own information clearly separated from everything else that is going on in the organization.

2.5.4 Component

Component is simply a wrapper for similar documents. All documents under a single component always use the same template. This term is rather rarely used as it is quite a technical concept. Sometimes components are also called tools.

2.5.5 Document

A single page in HIP is named a document. Therefore, regardless of what page is created in HIP, technically it is always a document. In everyday usage this term is actually quite rarely used, because if a new customer is created in a CRM system, the document is most probably called a customer, project document in PMS is most likely called a project and so on. This thesis does not focus on a single use case; however, it is good to know this term, because it will be used later on.

2.5.6 Template

Template (see Figure 5) defines the data structure and layout for documents. Otherwise it is like any other HTML page with HTML, CSS and JavaScript markup, but it is spiced with certain “namespaced” elements which are called widgets. The markup is passed to client as it is, but widgets are handled separately and they contain their own logic what content they write out to the client.

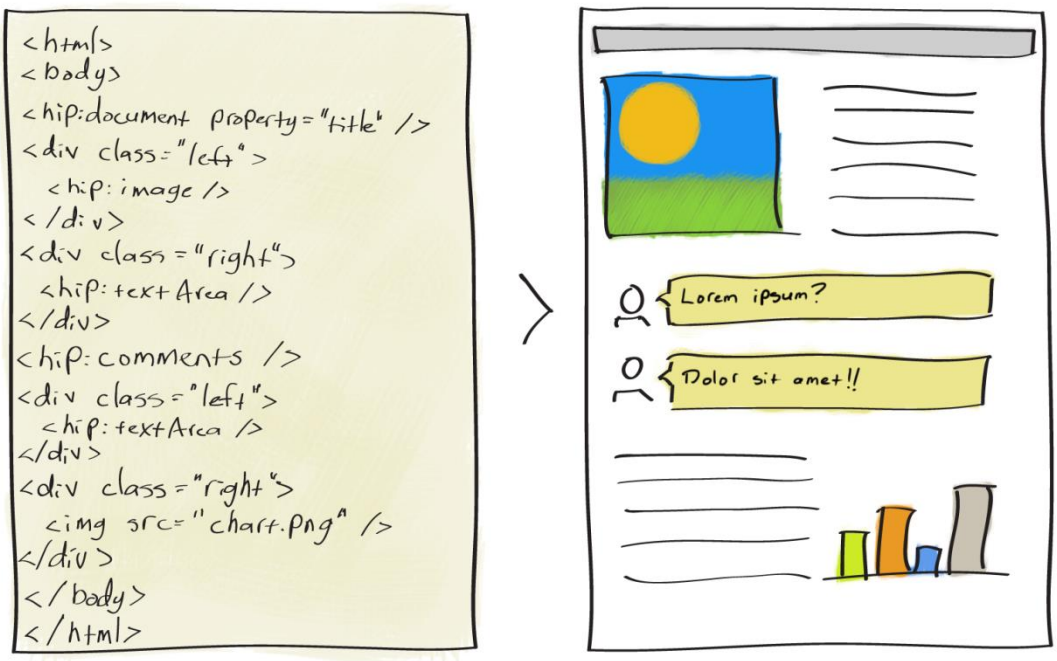

FIGURE 5. Visualization of template code and the outcome as document

2.5.7 Widget

A very good and compact definition of widgets can be found in Wikipedia which states that “widgets are basic visual building blocks which, combined in an application, hold all the data processed by the application and the available interactions on this data” (GUI widget 2013). In HIP these widgets are, for example, text field, text editor, date and time inputs, comments, relations, attachments and menu, just to mention a few. There is a vast amount of these widgets and each of them can be customized for different needs. New widgets or widget customization options can be easily added due to very modular architecture. The combination of widgets and basic markup language in templates has been proven to be a very quick and cost efficient way to build just the right tools for all kinds of needs.

2.5.8 What’s New

HIP has a feature named as what’s new, more commonly known to wider audience as news feed, which can be used to follow everything that is happening in a workspace. This list is personal and contains unread items since user’s last logout. Throughout the lifespan of HIP this list has been updated only when user enters a

workspace or when user updates what's new manually. In latest HIP version a real-time messaging system was implemented which can be used to update what's new list also automatically in real-time whenever something happens in the workspace.

2.5.9 JavaScript API

HIP has also an extensive JavaScript API which can be used to create highly customized implementations for unique needs. API supports various actions such as creating and searching documents, getting user information and reading and editing widgets' data. This means that even though widgets in HIP contain both encapsulated data format and user interface, it is possible to utilize just the data part and create completely customized user interfaces for different widgets. Via API information can also be collected from various sources to build, for example, bar charts or reports. The data can be just about anything collected inside or outside the system so mainly imagination is the only limitation what can be built utilizing the API.

2.6 Future Visions

HIP is being constantly developed and there are always new plans about the next steps in development of the platform.

Latest addition to the platform's core level features was real-time communication. Using this feature, real-time messages can be sent and received which enables, for example, updating workspace news feed in real-time giving user feedback immediately when something relevant is happening. It is also possible to update document information on the page in real-time. This means, for example, that it is possible to use comments similar to chat; however, the information is shared and preserved on a page, not private and lost like in chat. Real-time opens various possibilities and in the future real-time messaging will probably find its way into many features.

As the list of available widgets and widget configuration options is already quite extensive there rarely is a need to add new ones. Sometimes though there might be an encounter with a project in which something new is needed which either cannot be built using existing functionality or it would be more efficient to create a new widget.

In rare cases like these there is an option to add new functionality into HIP either in form of new widgets, API functions or potentially even some core level changes which affect all users regardless of their setup.

Good care has been taken of system architecture especially because severely bloated architecture was the main reason why the technological transition from Humap Tool to HIP had to be done. Nevertheless, keeping the architecture in good shape means constant refactoring and maintenance. Humap has a constant goal to keep looking the architecture from an as external point of view as possible and trying not to limit the visions according to what has been built so far.

In addition to good architecture, one of Humap's goals is to keep HIP as a robust and well performing platform. There are certain complex features in the core of the platform which are optimized continuously in order to keep the overall performance as good as possible.

3 Feature and Technology Researches

3.1 Reasons for Conducting the Researches

Trying to offer everything via mobile application that is available through desktop client rarely makes sense. Desktop users are in different mode. They have time to wait some dozens of seconds or minutes for their computer to start, their hardware generally performs better than in mobile devices and usually desktop users have bigger resolutions, keyboard and mouse or touchpad. Consequently, mobile users often have touch-based devices and smaller displays with smaller resolutions.

This is why it is very important to research and identify what parts of services should be available via mobile client and how they should be implemented. For example, in HIP there are numerous administrator features which are often rather complex and rarely needed operations compared to features which are used on a daily basis. Thus it would not make much sense to implement features like user management in mobile client as it is needed only occasionally by a very small percentage of users.

Two kinds of researches were needed before the design and implementation phase of the mobile client could be started. The first was purely technical research which was needed in order to find out the technologies that would help the company to develop as good mobile client as possible using minimum amount of time and effort. The other research was based on HIP itself in order to find out what kind of features should be implemented in the mobile client.

3.2 Research Methods

Technology research was done by first searching online for most used development platforms and frameworks for building mobile applications. This research was done by JAMK students, a group called Ifelse. They provided Humap a list that was investigated more closely in order to find out how the selected technologies work and what it would require to use them. Once certain properties of these technologies became clearer it was rather easy to drop some technologies out from the list and narrow the choices until the right setup was found. Responsive design was something that was already familiar as a term so Ifelse did not need to do extensive background check about that. Information was searched about responsive design online and other developers were asked about their opinions and past experiences.

Quantitative and qualitative research methods were chosen for researching existing HIP features. This was a rather natural choice as there were two major issues to be investigated. The first one was the raw numeric data about how much current features are used and quantitative methods fit perfectly to harvest this data into theories that would support the selection of the right features to be implemented into the mobile client (Quantitative research 2013). Secondly, Humap wanted to hear current users' opinions about what kind of features they would like to have in the mobile version of their current environments. For these purposes qualitative research method was used which in this case means a survey with one open question and ten Likert scale questions (Qualitative research 2013).

3.3 Technology Research

3.3.1 Collaborative Research

With technology research there were few main goals out of which the two most important ones were limiting the researched technologies and choosing the right people to work with. It was rather clear from the beginning that Humap could use some help to conduct the technology research. In a small company like Humap Software it can be hard to allocate required amount of resources for such a large task. Technology research was something that could have been done as internal work as well. However, without utilizing external resources it would have just taken more time to do it in between all the customer projects and other ongoing daily tasks in the company.

Timing could not have been any better for this thesis as students at JAMK University of Applied Sciences Degree Programme in Media Engineering were looking for partners for projects related to their studies. Once ideas for mobile technology research were given to JAMK, a student group named Ifelse was interested about the topic and soon a fruitful collaboration was started.

3.3.2 Preconditions

By the time Humap had the first coffee table discussions about the mobile client there was already a vast amount of mobile devices and operating systems available. First and foremost it was clear that binding development into just a single operating system or specific hardware was to be avoided. The mobile client was to support "just about all modern devices and operating systems". The main purpose was to create a single codebase which would work for most mobile users without having to maintain separate codebases for different devices or operating system.

Humap Software also has solid knowledge and a great deal of experience about traditional web technologies including HTML, CSS and JavaScript. From all the available options it was clear that solutions related to these technologies were preferred as they were familiar to the company and the company knew how to implement various solutions quickly utilizing these languages. In addition, current HIP platform client

side codebase is implemented using these languages and there is a principle to keep the amount of different programming languages in minimum. There was previous knowledge and some experience in other languages as well, such as Java, at Humap Software. Still, introducing a new language or in worst case multiple languages into the pool of HIP programming technologies would mean more overhead and as a small company it was to be avoided.

3.3.3 Responsive Web Design

One of the increasingly popular options for mobile implementations is responsive design. Responsive web design aims to create one single codebase which adapts to wide range of devices varying from desktop monitors to small smart phone displays. It uses CSS3 media queries to adapt the content to different resolutions and aspect ratios. Once a proper layout is designed and rules for different resolutions are written there is virtually no limit how the layout can scale to different sized screens. (Responsive web design 2013.)

Responsive design was something that was not delegated to Ifelse; however, this technology was investigated by Humap. It sounded like a tempting option as the same programming languages that have been used so far could be utilized. Also, responsive design aims at a single codebase so from this point of view responsive design fits the preconditions.

The core principle of responsive web design is to manipulate page layout according to CSS3 media queries. The World Wide Web Consortium (W3C) states as follows about media queries.

> *A media query consists of a media type and zero or more expressions that check for the conditions of particular media features. Among the media features that can be used in media queries are 'width', 'height', and 'color'. By using media queries, presentations can be tailored to a specific range of output devices without changing the content itself. (CSS3 Media Queries 2012.)*

A simple CSS example utilizing media queries looks something like this. First there are general definitions to the side panel which always apply and the side panel width is

set to 300 pixels and it floats to the right side of the screen. Then using media query there are separate definitions which apply when the display resolution maximum width of a device is 1000 pixels simulating roughly a tablet device in landscape orientation. For those devices the side panel is slightly narrower, but it still floats to the right. Finally, there is a separate block which applies when maximum width is 320 pixels which is a rather common maximum width for a mobile phone in portrait orientation. For those small screen devices the side panel does not float anymore, but instead it is placed to where it naturally fits which is most probably below the main content.

```
/* general styles for element with has class name sidePanel */
.sidePanel {
     width: 300px;
     float: right;
}

@media (max-width:1000px) {
     /* these styles apply when resolution is
     less than or equals 1000 pixels in width */
     .sidePanel {
          width: 250px;
     }
}

@media (max-width:320px) {
     /* these styles apply when resolution is
     less than or equals 320 pixels in width */
     .sidePanel {
          width: auto;
          float: none;
     }
}
```

It needs to be kept in mind that this is a simple example and there are endless ways how to design and utilize CSS media queries in order to create very flexible and fluid layouts. In order to grasp the idea how responsive design looks like in action Figure 6 and Figure 7 illustrate how a demo website (La 2011) looks rendered in different browser window sizes roughly mimicking desktop, tablet and phone display resolutions. Similarly as the CSS example the side panels will narrow with smaller screens

and eventually they will cease to float to right anymore. Instead, they will be placed below the main content and are available to user by scrolling the page down.

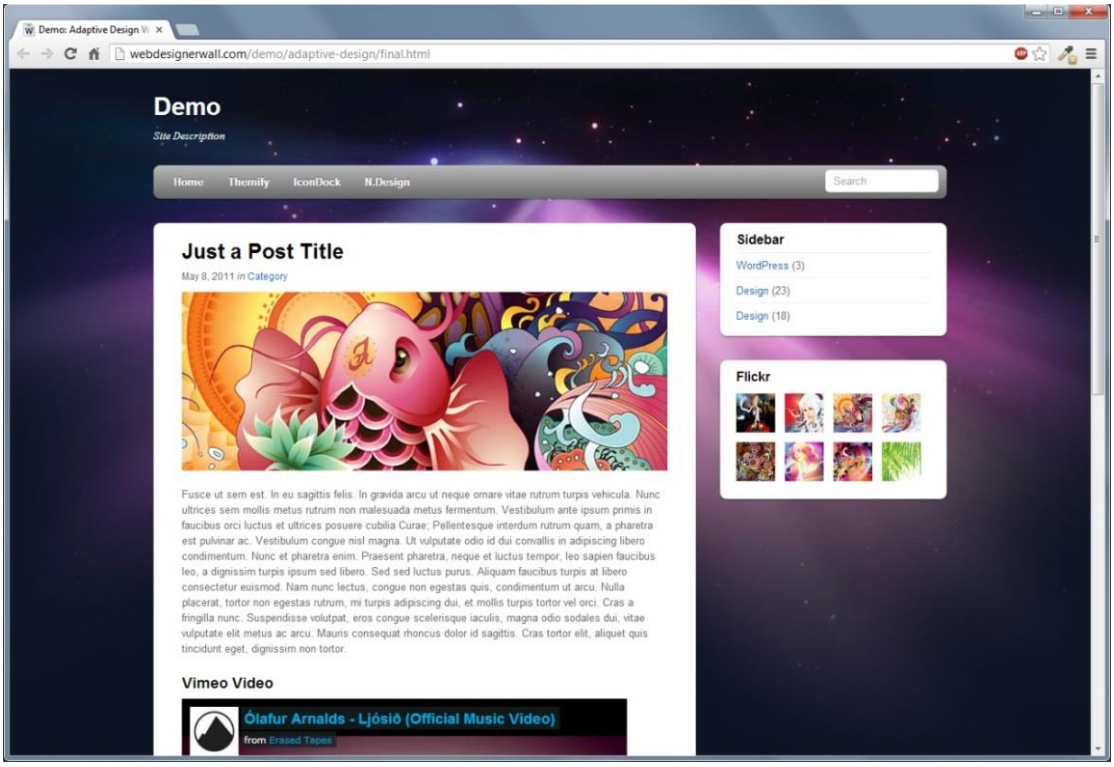

FIGURE 6. Desktop version of a demo website

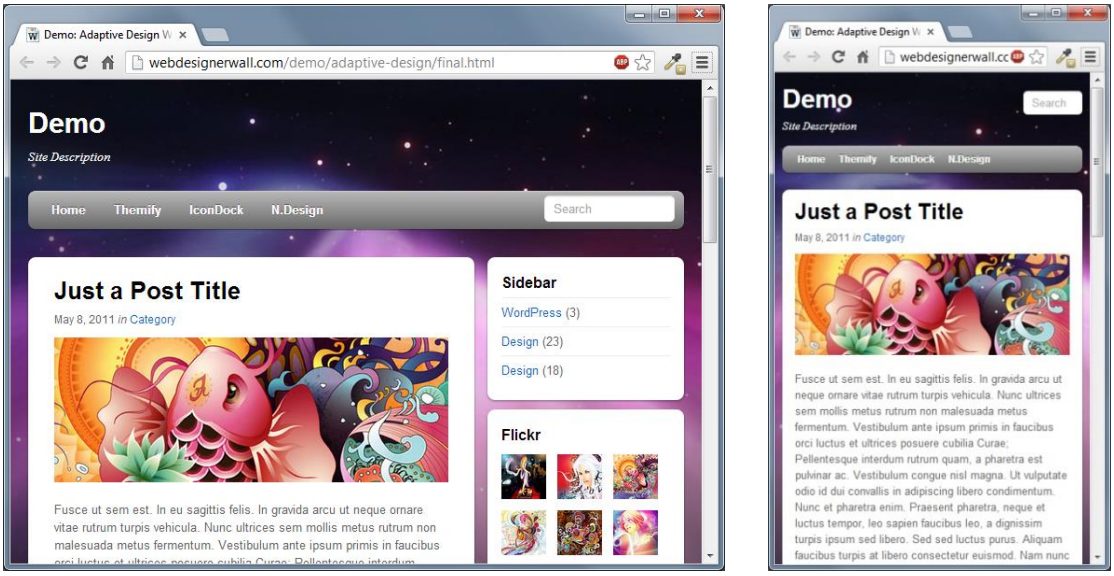

FIGURE 7. Demo website in tablet landscape and phone portrait orientations

Once responsive web design was studied more closely it was noticed that there were some issues in this approach. One of them was that the customer specific implementations are sometimes highly customized. Hence there are cases in which there is no

central place in the architecture to change code for the layout to become responsive. This is rather natural as different customers use very different layouts; thus there is no single rule how the layout should respond to, for example, different screen sizes.

There are also some other caveats to look into before rushing into responsive web design. Because responsive web design uses single codebase for both desktop and mobile devices it means that both environments have to download all the source code regardless of whether they need it or not. Especially for mobile devices this can be harmful as they have to download and execute code that is irrelevant for them. (Moth 2013.)

Images can cause some problems as an image optimized for desktop usage is probably too large for mobile usage. In comparison, an image optimized to mobile usage will not probably look good on large desktop screens. In case the image is just resized using CSS rules, the mobile client has to download the original size image and use its CPU to process the image into smaller size. This consumes both additional network bandwidth and CPU. It should be kept in mind that using CSS rule “display:none” just hides the image from display, but it does not keep the client from downloading the image thus consuming bandwidth. (Sarmiento 2011.)

Nevertheless, responsive web design is something that is further discussed in the thesis. There are some existing features in the platform that can be made to respond to different devices and therefore provide better user experience for users who will use desktop optimized version of HIP using a mobile device. For example, HIP shows most action buttons when the mouse cursor is hovered over an element. This lightens the layout as there is no need to show tens of different buttons on screen especially as users just read data way more often that edit something. Touch screen devices, however, have no mouse so they cannot hover over elements. Via responsive web design principles this case can be handled and action buttons can be unhidden making them clickable for touch screen device users.

Some discussions in LinkedIn User Experience group also suggest that developers should offer the desktop version of their site to tablet users instead of forcing them to use mobile version or suggest downloading a mobile application. This is because,

for example, a 10- inch tablet has adequate display size and resolution to comfortably display most desktop sites. One can argue though that tablet users and desktop users have different needs and these devices are used in different contexts. Anyway, it should be up to the user to decide whether to use the mobile client or desktop version of HIP as there are situations when both paths are justified. If mobile device users and especially tablet users choose to use the desktop version Humap aims to offer at least a usable version of HIP in the future by improving the platform's responsiveness.

3.3.4 Development Platforms

When the research on the options towards mobile client technology was started, the topic about existing development platforms and frameworks was a large unknown. Therefore, Ifelse student group was assigned to investigate what technologies, preferably open source, there are out there, how they work, and what would the student group see to be the best pick for HIP mobile client.

This list is not comprehensive covering all the possible development platforms. Instead Ifelse checked out some of the most common ones and filtered the following list for the company to choose from. First in the list there are three different development environments which are PhoneGap, RhoMobile and Titanium.

PhoneGap was originally developed by Nitobi Software, but was acquired by Adobe in 2011 (Adobe Announces Agreement to Acquire Nitobi, Creator of PhoneGap 2011). Soon after this more changes took place in the background of PhoneGap as Adobe/Nitobi donated the codebase to Apache Software Foundation (ASF) and it was named Apache Cordova (LeRoux 2012). This means that PhoneGap is based on Apache Cordova and is a distribution of it; however, it still remains its own free open source software.

PhoneGap is not actually that much of a development environment. It is a HTML5 application platform which can be used to create native applications using standardized web technologies. This means that developers can utilize their existing HTML, JavaScript and CSS skills to create mobile websites and then turn them into native

applications using PhoneGap. While Ifelse was doing this report PhoneGap required developers to setup an operating system specific integrated development environment (IDE) that was used to build the native application. Since then PhoneGap has improved so that a cloud based service Adobe PhoneGap Build can be used to build native applications to several platforms without developer having to setup any IDEs. (PhoneGap 2013.)

RhoMobile Suite is an application development platform provided by Motorola. It is a closed environment which consists of RhoConnect, RhoStudio and RhoElements. Similar to PhoneGap, the codebase is written using HTML5. (RhoMobile Suite 2013.)

Titanium is an open development environment created by Appcelerator. According to specifications it looks very similar to RhoMobile. The software development kit (SDK) is JavaScript-based and it provides loads of APIs for iOS, Android, Windows, Blackberry and HTML5. This means both native, HTML5 and hybrid applications can be built. They also provide their own ecosystem through Appcelerator Open Mobile Marketplace which can be used to purchase components to users' mobile applications. (Titanium Mobile Development Environment 2013.)

3.3.5 Frameworks

In addition to development platforms, Ifelse provided a list of frameworks that could be helpful in creation of the mobile client. Whereas the development platforms focus on converting the HTML, CSS and JavaScript code to native mobile application these frameworks have a different goal. They focus on helping developers to write the HTML, CSS and JavaScript code as quickly and easily as possible.

jQuery Mobile is HTML5-based framework that is built upon popular jQuery and jQuery UI foundation. For developers familiar with jQuery and jQuery UI there is minor or no overhead at all when transitioning to jQuery Mobile as all of them share very similar syntax and ideology. jQuery Mobile has a solid list of ready-made widgets and almost every mobile device is supported. Mobile platform support is categorized in A (full), B (full minus Ajax) and C (basic HTML) classes, but most mobile devices are included in category A. A full list of supported devices is available at jQuery Mobile's

homepage. What is also great about jQuery Mobile is that it is fully compatible with PhoneGap, so a website built upon jQuery Mobile can be later on built to native application very easily using Adobe PhoneGap Build. (jQuery Mobile 2013.)

Sencha Touch is a very similar framework compared to jQuery. Also Sencha Touch is HTML5-based, and it has a good list of ready-made widgets and is also compatible with PhoneGap. There are some differences between these two libraries though. Sencha Touch has a large object model which provides more features out of the box, but might take bit longer to learn than jQuery Mobile. Sencha Touch also targets less mobile devices than jQuery Mobile. Finally, Sencha Touch is very JavaScript focused and developers need to write very little HTML whereas in jQuery Mobile the library enhances HTML code utilizing techniques like HTML5 data attributes. (Sencha Touch 2013; Ramon 2011.)

Dojo Mobile, part of Dojo Toolkit, is yet another HTML5-based framework which is also very similar to jQuery Mobile and Sencha Touch. Mainly the difference to earlier two frameworks is the different kind of JavaScript syntax and different looking widgets. In addition, it allows users to utilize a very comprehensive list of features Dojo Toolkit which can be considered both pro and con as even if it can offer more functionality than the past two frameworks it can also take more time to study the ways of Dojo Toolkit. (Dojo Mobile 2013.)

EmbedJS has a different approach compared to the previous frameworks. Even it is based on Dojo Toolkit, it has just JavaScript API and no widget based system at all. Therefore, it is more similar to jQuery than to any of the listed frameworks so far. (EmbedJS 2013.)

The other two frameworks included in Ifelse group's report were Jo and Lungo. These two are both HTML5-based frameworks which are similar to jQuery Mobile, Sencha Touch and Dojo Mobile. They also differ just in terms of API syntax, user interface and list of available widgets. (Balmer 2012; Jimenez & Olalde 2013.)

3.3.6 Chosen Technologies

It was rather natural choice for us to go for jQuery Mobile. This was because HIP already uses both jQuery and jQuery UI so they are both familiar libraries to us. As the jQuery syntax has been used for some time it was noticed that jQuery Mobile looks very familiar and it was possible to build mock-up mobile website very quickly.

Moreover, jQuery Mobile theming is very easy using a tool called ThemeRoller (ThemeRoller For jQuery Mobile 2013). Developers can use ThemeRoller to easily drag and drop colors to UI swatches and then the theme can be downloaded as a zip file. The colors can be also chosen from Adobe Kuler color themes which is a service where users can create, share and rate color themes (Kuler 2013). By using Kuler color themes developers can easily pick colors which work harmoniously together. Still, even without creating custom themes the jQuery Mobile default theme looks nice and modern.

From the other frameworks Sencha Touch seemed most promising, but in the end it did not provide any reason to go for it over jQuery Mobile. Although both frameworks provide virtually same kind of results, Humap is familiar with jQuery syntax whereas with Sencha Touch developers would have had to learn how the framework works and what the best practices to use it are. According to the past experience, Dojo Toolkit has a comprehensive set of features; however, as for HIP, it was found to be bit too massive and complex to use. In addition, the layout of widgets was not as good as in Sencha Touch and jQuery Mobile. EmbedJS does not provide the right tools for this case as it does not have an easy-to-use widget list. Besides, it is based on Dojo Toolkit so leaving EmbedJS behind was probably the easiest choice. Finally, Jo and Lungo are very similar to jQuery Mobile and Sencha Touch and neither one of them provided any reason not to go for jQuery Mobile. They are somewhat less familiar to a large audience and there were some worries whether there is a large enough community behind those frameworks and how actively they are developed. Jo was being developed by just one author, Dave Balmer Jr. and Lungo by two authors, Javi Jimenez and Ignacio Olalde. With jQuery Mobile there are no worries about that as jQuery is a very popular and actively developed framework.

From the development platforms PhoneGap was easily the right choice for this case. There were multiple reasons why PhoneGap was chosen:

- It is simple and easy to use.
- PhoneGap was acquired by Adobe so it has a large company behind it.
- The only thing that worried Humap before was that PhoneGap Build required setting up platform specific IDE, but this issue was solved when Build turned into cloud-based service.

However, it was decided that development can very well begin with HTML5-based mobile application. Because HIP is already used as web service there is virtually no overhead to start providing a mobile version in some other URL. This would mean that the initial version of the mobile client cannot access mobile device native features such as camera or notifications. Later on, when the time is right to start developing such features in the client that require native features, a native version of the application can be built using PhoneGap without having to rewrite the existing codebase.

3.3.7 jQuery Mobile

At the moment jQuery Mobile is at version 1.3.0. It is recommended to use jQuery Mobile sources through CDN, but developers also have the option to download and host the files themselves. Using jQuery Mobile via CDN is very easy and requires adding only following lines in HTML code. It is to notice that these URLs are version specific and it is highly recommended to visit the homepage of jQuery Mobile at http://jquerymobile.com/ for up-to-date information instead of copy-pasting the following lines of code.

```
<link rel="stylesheet"
href="http://code.jquery.com/mobile/1.3.0/jquery.mobile-
1.3.0.min.css" />
<script src="http://code.jquery.com/jquery-1.8.2.min.js"></script>
<script src="http://code.jquery.com/mobile/1.3.0/jquery.mobile-
1.3.0.min.js"></script>
```

Once these files are included in the sources building the actual application HTML code can be started. Figure 8 shows a simple demo application built using simple drag and drop Codiqa application at jQuery Mobile home page.

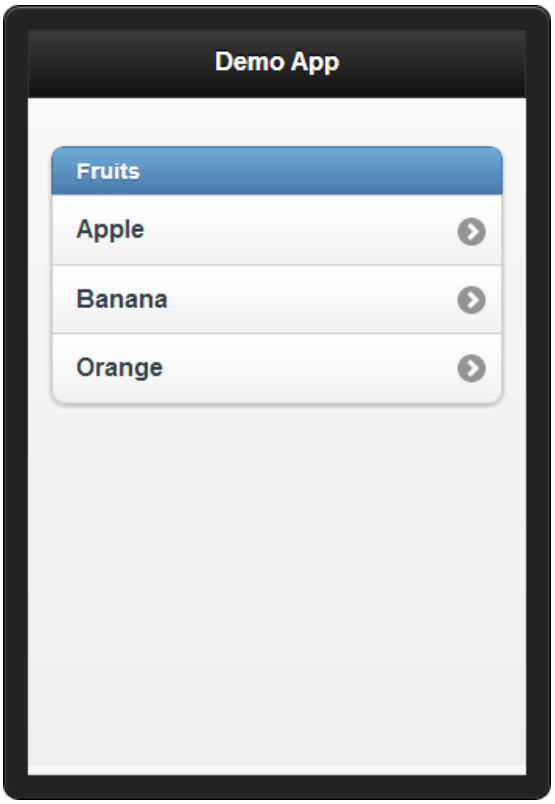

FIGURE 8. Simple demo application built using jQuery Mobile

This demo application shown in Figure 8 requires rather minimalistic amount of code. There is only HTML code needed and when spiced with proper data attributes jQuery Mobile turns certain elements into widgets such as header and interactive list view widget in this demo application. Here is the code generated by Codiqa.

```
<!-- Home -->
<div data-role="page" id="page1">
     <div data-theme="a" data-role="header">
          <h3>Demo App</h3>
     </div>
     <div data-role="content">
          <ul data-role="listview" data-divider-theme="b"
          data-inset="true">
               <li data-role="list-divider" role="heading">Fruits</li>
               <li data-theme="c">
                    <a href="#page1" data-transition="slide">
```

```
                    Apple
                </a>
            </li>
            <li data-theme="c">
                <a href="#page1" data-transition="slide">
                    Banana
                </a>
            </li>
            <li data-theme="c">
                <a href="#page1" data-transition="slide">
                    Orange
                </a>
            </li>
        </ul>
    </div>
</div>
```

jQuery Mobile offers also JavaScript API which can be used to enhance and extend these widgets and mobile application in general. Full API documentation and widget reference are available at jQuery Mobile's homepage.

When doing development using jQuery Mobile I really recommend visiting these links over reading books about the topic. When I first started to get into jQuery Mobile the book jQuery Mobile (Reid 2011) made very useful reading. Though it is informative book and gave a good overall picture and first sight into jQuery Mobile, the framework is developing at such rapid phase that the book is badly outdated. By visiting online resources up to date information can be always obtained.

3.3.8 PhoneGap

There are two ways to use PhoneGap: the easy way and the hard way.

The easy way is to use Adobe PhoneGap Build, a cloud-based service available at https://build.phonegap.com/, and let the cloud compile to various platforms. Developers do not need to setup any IDEs themselves, but just either upload the code into Build as zip package or use Github repository and connect it to the Build account. Once Build has completed compiling the application it provides a QR code which can be scanned with mobile device in order to download and install the application for

test purposes. Free version of Build account can be used to build unlimited amount of open source applications and one private application. In case more private applications are needed developers have to pay a license which currently costs 9.99 US dollars per month.

Personally the author was really impressed by Adobe PhoneGap Build. Once the HIP mobile client prototype was built it was uploaded as a zip package into Build and in just some seconds a list of applications for different devices was provided. The Android version was downloaded by scanning the QR code and after changing the mobile phone security settings to allow installing applications outside Google Play the installation of the application was fast and easy. The HIP mobile client prototype did not work out of the box as native application as there were some relative URLs in it, but that does not change the fact that Build itself worked like a charm. Eventually it was a really quick operation to fix the codebase to work also as native application.

The hard way is to download PhoneGap and setup platform specific IDEs. Then projects in development environment need to be set up and platform specific native applications built one by one. This enables developers to have more control over the development process, but setting up various IDEs might take some time and when application code changes all supported platform specific native applications need to be built manually whereas Build does all the work for developers. Setting up IDEs also provides developers an option to test their applications using various emulators. This is especially useful in case developers do not own all the different devices that need to be supported. Also, when setting up the IDEs there are no worries about how many applications can be built for free.

If, and probably when, the application is to be deployed to platform specific store there is a need to check out the store specific documentations. Apple App Store for iOS apps, Google Play for Android apps and Windows Phone Store for Windows Phone applications all differ in terms of how to publish the applications.

If there is no need for mobile device native features in the application building it with PhoneGap does not require anything special. The HTML5-based code can be built as native application and that is it. In case native features such as camera are to be uti-

lized, a proper plugin needs to be included in the application. If using Adobe PhoneGap Build config.xml file can be used in the root folder of the project and he following line is to be added to it in order to include camera support in the application.

```
<feature name="http://api.phonegap.com/1.0/camera"/>
```

There is more documentation available about how to use config.xml file with Adobe PhoneGap Build at https://build.phonegap.com/docs/config-xml. If using the own IDE then there are platform specific ways to enable camera functionality in the application.

Developers are recommended to take a look at the API documentation available at http://docs.phonegap.com/. This documentation covers the usage of different plugins and examples how to access these native features in the application's JavaScript code.

Below is a quick example how to take a photo using mobile device's camera and show this picture as an image element in the application.

```
// getPicture function parameters are success handler function,
// failure handler function and configuration object
navigator.camera.getPicture(
     function(imageData) {
          var imageSrc = '' + imageData;
          document.getElementById('myImage').src = imageSrc;
     },
     function(message) {
          alert('Could not take picture because: ' + message);
     },
     {
          quality: 100,
          destinationType: Camera.DestinationType.DATA_URL
     }
);
```

3.4 Feature Research

HIP has been developed for five years now, but it has roots in Humap Tool so in some sense one could say it has been developed for 13 years. During all those years the

platform has grown to have hundreds of features and squeezing them all or even a major part of those features into mobile client would be just massive waste of time. Because of this it is extremely important to carefully research what are the features that users really need in the mobile client.

3.4.1 Analyzing HIP Logs

There are two kinds of log files which HIP collects.

HIP uses Apache HTTP server which keeps its own access log containing information about all requests made to Apache. This log file contains a load of information including requests to all HTML, CSS, JavaScript, image and other files served via Apache. This kind of information can be difficult to filter into such format from which details such as how many times certain HIP features have been used could be separated. Analyzing access logs every once in a while can be very useful from development and optimization point of view though. One can find out obvious issues like broken links, images, JavaScript or CSS file includes from repeated 404 errors in access log. Furthermore, access log can be used to see different kind of usage patterns and to find out, for example, if the service has been attacked against from certain IP address or range of IP addresses.

Because access log contains such a vast amount of information there is a secondary log which is the platform's own action log. It contains detailed information about what user did and when this action was done; however, it logs only HIP related actions and there are some actions in HIP that are not logged into action log. For example, different static HTML, CSS, JavaScript and image files served through Apache are not logged into the platform's own action log. This log gives a good overall picture about which HIP actions are used the most. HIP action logs analyzed in this thesis are collected from period starting from 2nd of April 2009 until 1st of November 2012.

The most interesting fact was to know the relation between reading and editing data in HIP. The result was rather clear – users read data about three times more often than edit it just in terms of raw data as can be seen in Figure 9.

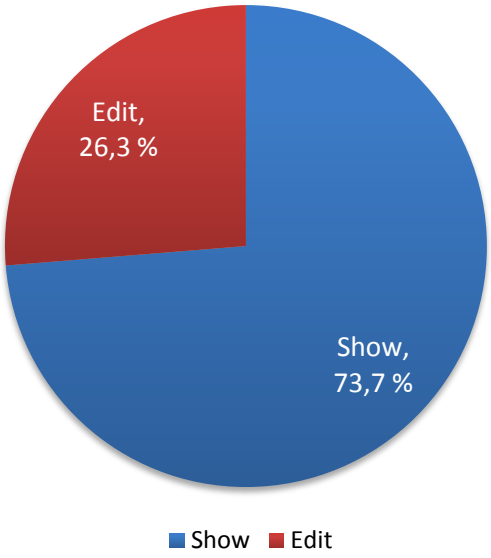

FIGURE 9. Amount of action log show actions compared to edit actions

It is also known that reading a document logs just one entry in HIP action log, but editing document data can be done per widget. This means that there can be various log entries when user edits, for example, project details such as project status, deadline, specifications, attachments and comments without having to reload the page in between. In addition, listing documents such as projects is not logged as an action. Sometimes users can find the data they were looking for already in the documents list view (see Figure 10) which does not log any read actions, but can show simple inline information like short text, numbers, status, users and dates. This also reduces the amount of show actions in the action log though in reality users are reading data even more than what is logged. Therefore it is safe to say that in HIP reading data is a more popular action in comparison to editing data.

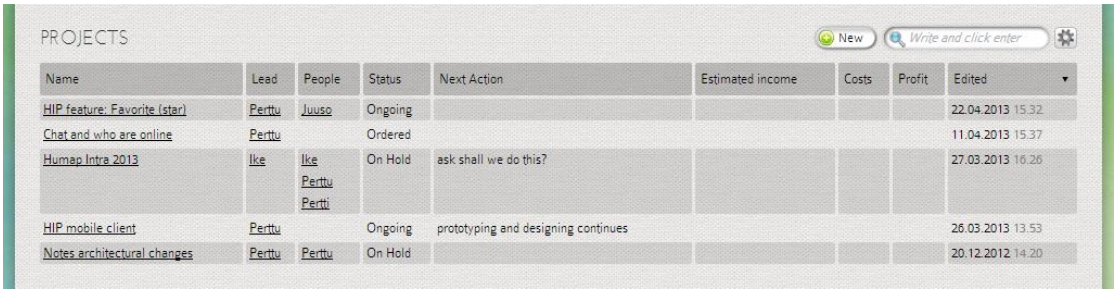

FIGURE 10. Part of HIP list page from Humap Software intra

This is one indicator that the first priority in the mobile client should be data availability. Users log into the system primarily in order to find data and creating or editing the existing data is a secondary need. This also gives some tips towards system performance optimization. Enhancing read action performance seems to have more overall performance benefits in comparison to edit action performance enhancements.

For the secondary need to create or edit data one has to dive a little deeper into the log files and separate the edit actions. Furthermore, these edit actions are separated into creating new data and editing or deleting existing data. Editing seems to be clearly the winner here owning almost 70% of all edit actions (see Figure 11). There is a catch here though which is that when a document is created it will contain empty widgets. So the logical order of actions is to create a new document which logs as one entry and then users will most likely input some basic information such as project or customer details, which usually sums up to multiple edit actions. In HIP action log there is no easy way to separate information whether widget was edited for the first time, in other words whether data was created, or was a user editing a non-empty widget, which means editing existing data. Nevertheless, creating documents and editing data are the clear winners here as deleting existing data is done very rarely.

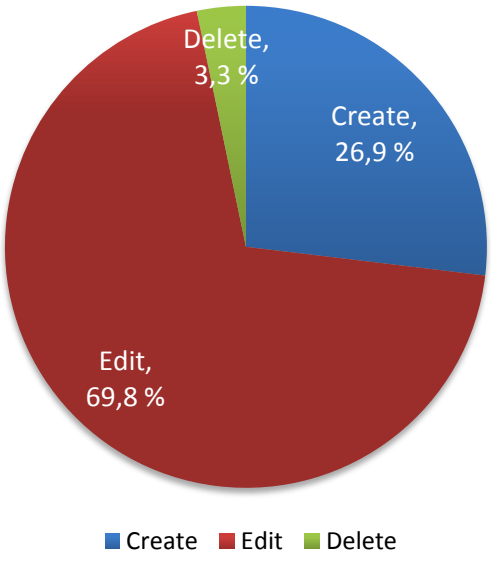

FIGURE 11. Edit actions separated into create, edit and delete entries

Document creation is an action that is already as fine-grained as it gets; it cannot be split into smaller categories. But when it comes to editing data there are some interesting facts that can be looked into. The six most edited widget types are comment, text, file, relation, tag and date which can be seen in Figure 12.

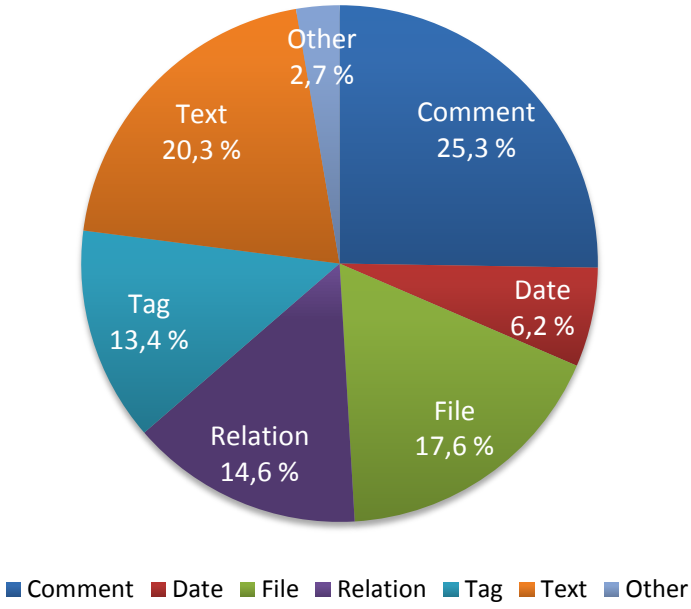

FIGURE 12. Edit actions split between six most used widgets

This information about the popularity of widget edit actions can be used to prioritize tasks in mobile client. Writing comments is the single most popular action in HIP, but not by a huge margin compared to editing text and uploading files. Relations are used to connect similar documents to each other and tags can be used to give users a predefined list of options that can be used to specify certain meanings for documents. Only date is clearly left behind from the other popular widgets as only about one of fifteen edit actions is used to edit dates. All other widget edit actions combined share just 2.7% of all edit actions in the action log. If functionality to write comments and edit text in the mobile client are implemented it already tackles almost half of the data editing needs. However, these numbers are from the desktop version so too hasty conclusions should not be made. Nevertheless, this information gives some data to work with when considering what features to implement in the mobile client.

3.4.2 Web Survey

In addition to analyzing log files, HIP users and closest partners were asked about mobile client features via web survey. This survey contained ten Likert scale questions on scale 1-6 and one open question. On the scale 1 means “not useful at all” and 6 means “very useful”. The scale was intentionally set to even count so that there is no middle choice to choose from as in “I do not know”. Users were forced to make a statement either for or against some feature. In the scale 3 means “maybe not so useful” whereas 4 stands for “might be useful”.

Unfortunately, very few people invested few minutes to answer the survey and only twelve respondents completed the survey. Therefore these results do not give valid data about what HIP users really thought about oncoming HIP mobile client. Nevertheless, there were few answers and they gave some direction with the mobile client design process. The response averages are shown in Figure 13.

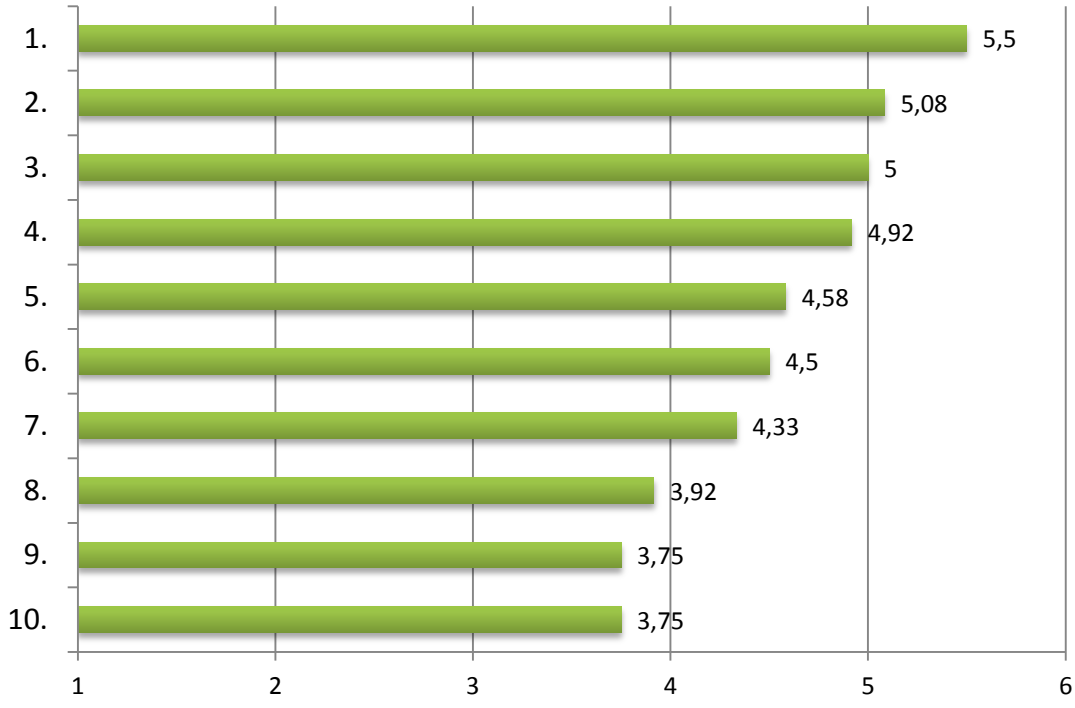

FIGURE 13. Questions ordered by averages from most to least voted

Below is the list of questions numbered in the same order as in Figure 13.

1. Possibility to filter what's new flow based on workspace, certain tools or certain documents (for example, follow and/or receive notifications from those documents only which you are somehow connected to)
2. Possibility to add comments to documents
3. Possibility to follow workspace's what's new flow
4. Possibility to save photos to a document using mobile device's camera
5. Possibility to create new documents
6. Possibility to edit some information in document (for example, change project's status or deadline, edit document's relations to other documents)
7. Possibility to save a video to a document using mobile device's video camera
8. Possibility to draw or sketch a picture or a chart to a document using mobile device's touch screen
9. Possibility to get notification to mobile device when something happens in a workspace
10. Possibility to record speech to a document using mobile device's microphone

There are some questions that are bound to each other. For example, what's new flow cannot be filtered in case users cannot follow the workspace's what's new flow in the mobile client. Therefore, it makes no sense that filtering was voted as number one feature whereas following what's new flow was the third most voted feature in the list. Nevertheless, it gives a signal that responders saw it as a really important feature to be able to filter what's new flow instead of just following everything that is happening in a single workspace.

When pondering the reasons for this it is rather natural. Mobile devices are personal and almost always with you. In case the mobile client would inform you with notifications when something has happened in the workspace it would not require a big team of users to make it a really annoying feature. The phone would just keep on beeping about something happening in the workspace no matter how irrelevant this information would be for a user. Consequently, notifications were second least voted feature in the list. In addition, desktop version of HIP did not have any real-time features by the time this web survey was active so people did not have any experiences

about real-time usage at that time. Now that desktop version what's new flow can be updated in real-time it might change the way users perceive the what's new flow. In the old version what's new updated only when user logged into a workspace. That is a completely different mode of using a workspace.

The respondents saw commenting also as a very important feature in a mobile client and it was the second most voted feature in the survey. Writing comments is a quick, easy and light-weight operation to be done using limited writing capabilities of mobile devices which is usually a virtual keyboard. More demanding document edit operations were 6th most voted feature so clearly people see writing comments to be a much easier operation than editing other widgets in documents using mobile device.

What speaks strongly for native application is that taking a photo using mobile device's camera was 4th most voted feature. This gives a reason to really consider making a native application rather soon in comparison to just settling to HTML5-based version of mobile client.

The open question in the survey was "what other features besides the ones above you would see relevant to have in HIP mobile client". Some respondents wrote more lengthy responses, some just few lines and some answered if the question would have been "anything else you would like to add". What was great though was that every respondent wrote at least something to the open question. Single statements were extracted from all of the answers and sorted out in a way that made it possible to comment on similar answers as a group. Similar answers are listed as a group and that is followed by an analysis of following questions.

- Simple and excellent usability for all features.
- More minimalistic layout compared to desktop version.
- In which situations mobile is chosen, for example, instead of laptop? When would it be relevant for me to use small screen instead of big display?
- I am not sure how relevant it is to be able to create and edit documents in mobile client, remember to keep it simple.

First and foremost, "simple and excellent usability" is something that Humap really tries to head towards. Trying to make a more minimalistic layout in mobile compared

to desktop version is not something to settle for. Mobile world differs from desktop so drastically that this work needed to start from scratch digging into the very basics and this is exactly why the survey and log analysis were done. It is nice to see that respondents were also wondering in which situations using mobile client is relevant over using a desktop version of HIP. Finally, “remember to keep it simple” is an excellent rule of thumb to be kept in mind, not just throughout the process of developing a mobile client, but for just about any software project.

- Will the information in mobile client be secured?

Data security is a valid concern and the same methods are used in mobile client as in desktop version such as SSL encryption to ensure that data will be secured. It is important to let users know how their data is protected as this seemed to evoke concern even though there were just twelve respondents in the survey.

- For me it feels like the most important thing would be to have access to all data in the system regardless where I am.
- Tablets are increasingly popular and are often used in trains, busses or hotels so it might be relevant to take those also into consideration when developing the mobile client.
- Camera and video features sounds like a nice thing to have in certain cases. I could imagine it would be handy to take a photo of some process chart, then share and comment those in HIP.

The list suits well with what has been analyzed in this thesis so far. The log files revealed that reading data is a more common action than editing data. Therefore, the assumption made by one respondent that it is important to have access to data is in line with these results. Tablets indeed are increasingly popular and bit different devices compared to smart phones so it is important to keep also tablets in mind throughout the development process. Taking a photo using mobile device was the 4th most voted feature in Likert scale questions. It is easy to imagine that it is a rather common action to draw some ideas or process charts on a whiteboard or a flip chart. It might be really handy if one could just snap a photo from the sketches and easily share these results in HIP for other colleagues to see and comment. The simpler the

process the more likely it is that such feature would be used. Users and especially mobile users are usually not ready to invest a lot of time into doing operations that should be simple.

- What kind of calendar features might there be in the mobile client?
- Microblogging, for example, setting your personal daily status.
- Showing your location based on HTML5 geolocation.
- Work time tracking (recording and/or input afterwards).
- Inputting comments or other long texts to system using mobile device's speech to text -feature.

This group of answers can be put to the product backlog as ideas and implementing them can be considered once the initial version is done. Calendar features need to be evaluated carefully. They might provide added value; however, according to Humap's past experience working with calendar features can easily require a great deal of time. This is because there are loads of different applications used and calendar sync can be a tough nut to crack properly. In the modern world the information flow can be quite overwhelming thus having automatic reminders could be a really important feature.

Microblogging is something that could be done utilizing comments in mobile client. However, it might be relevant to consider how to make it as easy as possible so that users do not have to navigate to a certain document before one is able to, for example, set personal daily status.

HTML5-based geolocation is also technically a rather easy feature to do. Some issues to consider would be how to use it and why.

Some simple work time tracking requirements have been solved in the desktop version of HIP. By learning from those experiences Humap believes they could provide even better work time tracking possibilities in the mobile client compared to desktop version.

Speech to text is available at least in Android devices in every input. Other devices could be investigated and mobile client could aim to provide as easy as possible way

for users to activate speech to text -feature. Especially when using a smart phone in portrait orientation while on the move it can be quite tricky to use a virtual keyboard for writing long comments. Conversely, it could be very easy to speak out ideas and one's mobile device would transform that as a written comment in HIP.

3.4.3 Putting It All Together

Mobile technology and feature research provides a long list of tools to be used. Out of all those results a game plan was crafted and here are the blueprints how the project will proceed with the development of HIP mobile client.

The first version of mobile client will be HTML5-based application built using jQuery Mobile. Features for reading will be prioritized so in the first version users can read workspace news feed and search for documents using simple search functionality. Doing some navigation features is also considered, but going through all workspace documents without using search is probably a rarely needed feature. Mobile users are more specific and it can be assumed they are not after browsing through long lists of documents.

After the initial version widgets will be made editable. Here widgets will be roughly prioritized in the same order as how popular their edit actions are in desktop version according to action log. Consequently, comments and text widgets are the first two in line. The workload will also be considered. How easy it is from technical point of view to implement something might change in which order certain widgets are made editable.

Once the first version is stable and in production the next step is to start considering adding features that require mobile device native features. The first one in the list is definitely taking a photo using mobile device's camera. Even though there were not many respondents in the web survey this feature was voted quite high and the use case provided in open question seems like something that could very well be a common need in various environments. After the first working native application is built it is not a major issue to add more native features such as notifications into the mobile client.

4 HIP Mobile Client

4.1 General Structure

The general layout of the mobile client is rather simple. Layout has a header which contains few navigation buttons which are accounts, what's new and search. The rest of the area is reserved for content which changes according to what a user is doing in the client. Browsing backwards in history can be done using mobile device's own back button. There are various examples of the layout in the following chapters which illustrate how the application looks like.

4.2 Authentication

Authentication to HIP is done by providing either user's email address or username and password. The server then searches whether a corresponding user is found by email address or username. Once the user is found the provided password is encrypted and compared against the one found in the database. If the password is correct then the user is authenticated, in any other case authentication fails.

In mobile client the situation becomes technically somewhat more complex. Users rarely want to repeatedly input username and password. Instead, once proper credentials are provided they are often securely stored in mobile application. Thus, users do not need to input credentials again in the future.

In HIP mobile client this becomes increasingly important. There can be cases in which users have various HIP user accounts. They would probably get tired with the mobile client rather quickly if they would have to input certain credentials, then logout and input other credentials to checkout another HIP based service.

Current solution was inspired by Google's multiple sign-in. Multiple sign-in makes it possible to sign-in onto multiple accounts and then switch between the accounts easily. In HIP mobile client there is a button in header navigation labeled as accounts. This button takes users to a view which lists all previously added accounts. A single item in the accounts list consists of user's picture, email address and the URL in

which the service is running. At the right side of the item there is an arrow symbol which indicates that when the item is clicked users will be redirected to another view.

The active account is marked with a symbol and can be changed by simply clicking one of the accounts. This view is then immediately followed by another view in which a user selects the workspace to be used (see Figure 14). After selecting a workspace the user will be redirected to what's new view that will contain the data from the selected workspace of the selected account. Again, the arrow symbol in the list indicates that by clicking an item on the list user will be redirected to another view.

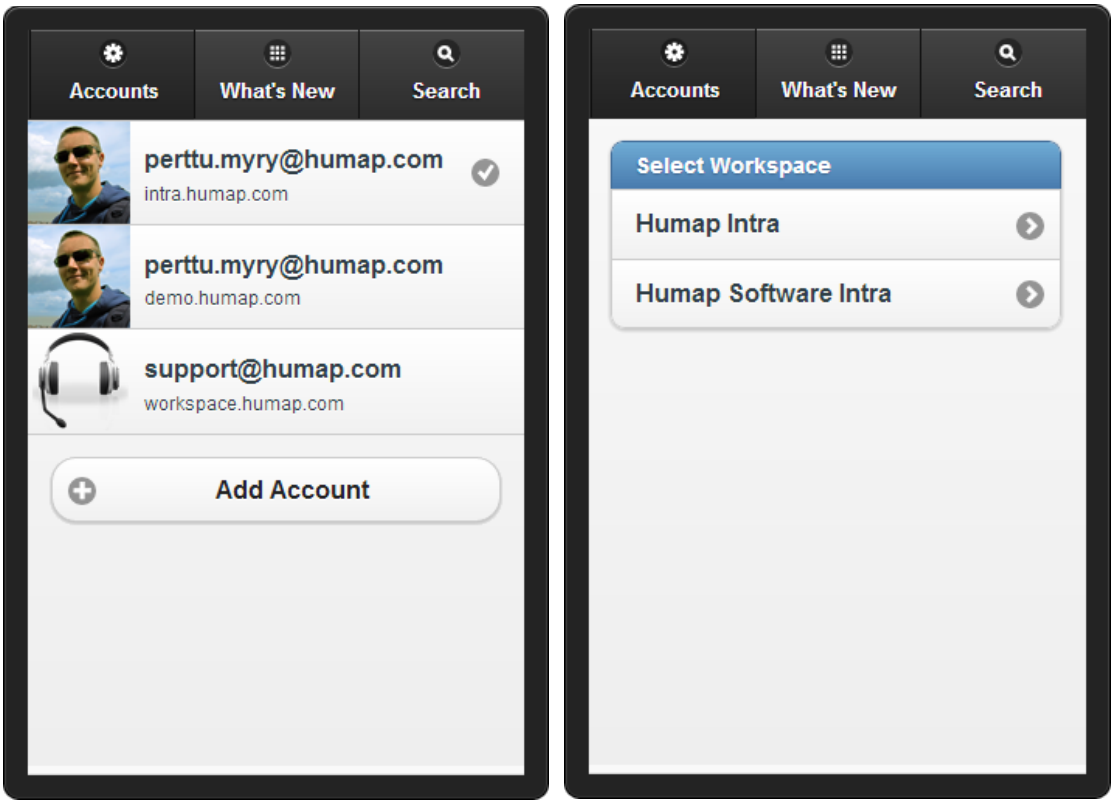

FIGURE 14. Accounts and select workspace views

Users can also add new account from accounts view by clicking the add account button at the bottom of the view. When adding a new account users must provide the URL such as intra.humap.com in which the service is running combined with email address and password which will be used to authenticate the user (see Figure 15). After clicking add button the account details are verified. If all information is correct user will be switched back to accounts view and the new account is added to the list.

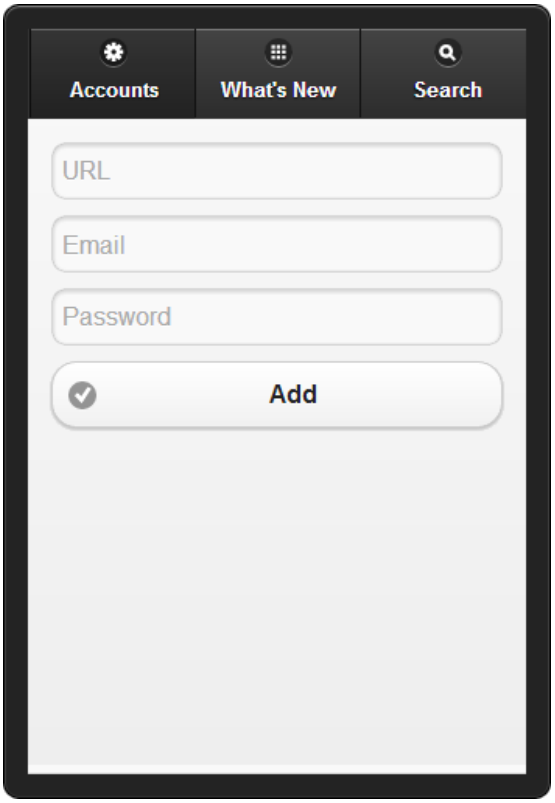

FIGURE 15. Add account view

4.3 What's New

News feed in HIP is called what's new and it can be accessed by what's new button in the header navigation. This view (see Figure 16) shows users a personal list of activities that took place in a certain workspace since their last visit.

The view contains list of documents which have any modifications that are unseen by the user. The single document row consists of few items. First there is an image of the author of the latest modification in that document. Then the row contains the title of the document and optionally a short highlight of what was changed. This highlight is not shown for all actions such as uploading a file or adding a relation to another document.

On the right side of the row there are two symbols. One is a balloon showing the count of unseen changes in the document and the other one is an arrow icon. Once a user clicks an item in what's new list the user interface will switch into corresponding document view. Now that user has seen the changes this item is no longer unread and will be removed from what's new list.

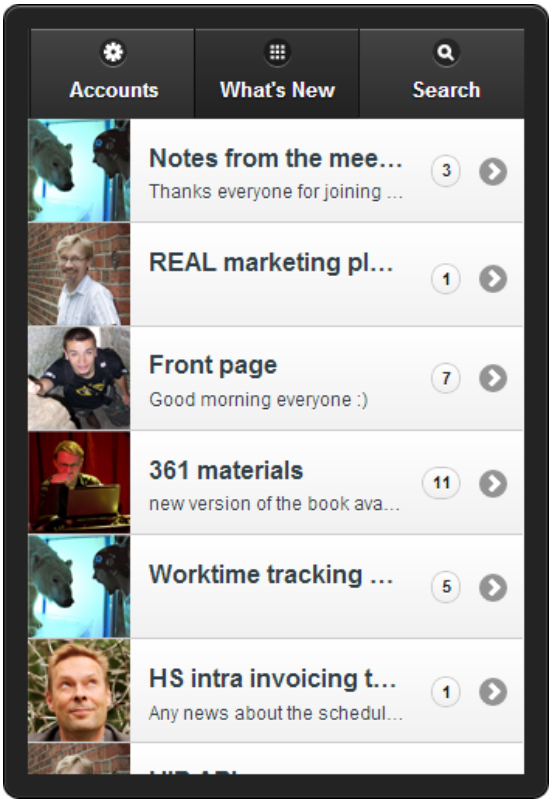

FIGURE 16. What's new view

4.4 Search

Search view can be accessed from the search button in header navigation. It can be used to search for any particular string in the selected workspace of the selected account. By default this view contains just search input at the top of the page. Once user enters search expression and clicks enter, go or similar button in mobile device's keyboard or virtual keyboard, client will show a loading icon and start searching for corresponding documents. Once search is completed the search results will be shown in accordion UI categorized per component (see Figure 17). User can clear previous search by clicking X button at the right side of the search input.

Each accordion item will contain the title of the component and a bubble in the right side which indicates the amount of found documents in that component. Clicking the accordion item will expand that item and collapse all other items. Documents in an opened accordion item are listed as links. Clicking a link will take user to corresponding document view. The reason why list view (see Figure 14 and Figure 16) was not used is that accordion itself is a list item. Showing document links as list item would be a list inside list which can look confusing.

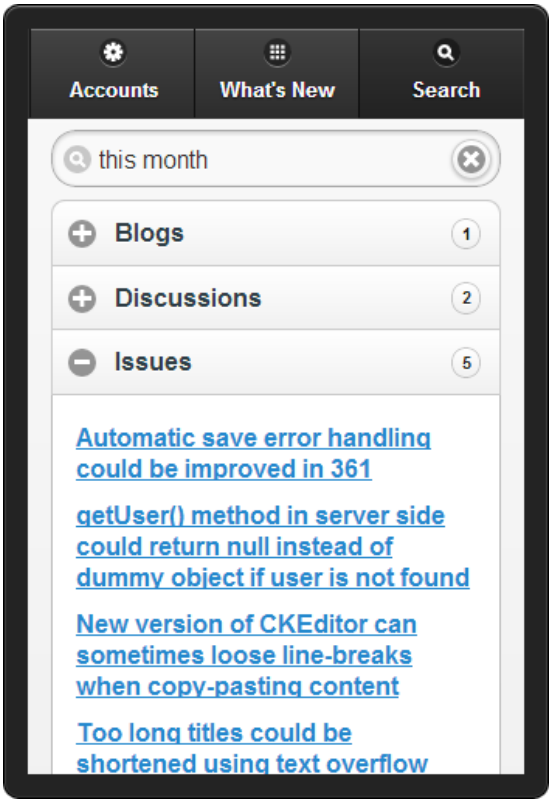

FIGURE 17. Search view

4.5 Document View

In mobile client document consists of only widgets. The regular HTML, CSS and JavaScript code included in the template is not processed in the mobile client at all. Only widgets are listed as accordion UI. By default all widgets in the accordion are open. Document list accordion differs from search results accordion so that each accordion item can be opened and closed separately. Furthermore, if a widget contains unread information for current user, the accordion item will be highlighted with different style to indicate that something has changed in this widget since user has visited the document (see Figure 18).

Different widgets have different looking content. The most basic widgets such as text field, tag list or date contain just textual and numeric information so those are displayed as text. Then there are some widgets such as relations and attachments which contain links to other documents or files in system. These contents are displayed as links in a similar way as in search view (see Figure 17). Image widget content is an image uploaded by user and the image is resized into mobile friendly resolution. Comments are shown as similar comments UI as they are shown in desktop version

of HIP. Comments are also capable of showing not just unread widget but also unread comments. This way users can see even more accurately what content is new since their last visit.

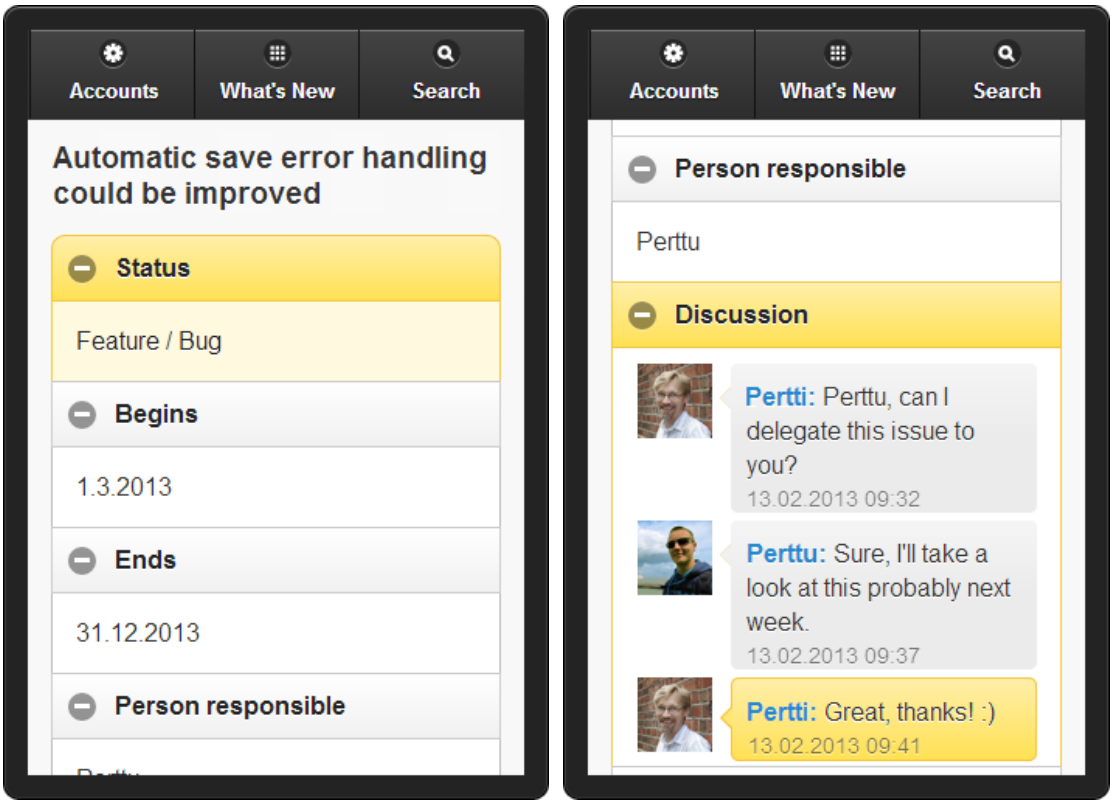

FIGURE 18. Document views

4.6 First Time Usage

When user opens up the mobile client for the first time there are no accounts saved into the client yet. When there are no accounts the client will go into first time usage mode. The first view that user will see is the add account view (see Figure 15). Once an account is added user will not be taken to accounts list, but instead directly to select workspace view (see Figure 14). Once user has selected the workspace the client will continue working normally and user will be redirected to what's new view.

4.7 Future Features

It was decided that the first version of the HIP mobile client is released for read-only purposes. Currently users cannot edit any contents or create documents by using the mobile client. This decision was backed up by the research results and due to tech-

nical reasons. It was somewhat easier to make the client just read data in the first version and leave editing and document creation features into the future.

It was also decided to go with HTML5-based version of HIP mobile client in the first step. This means that the first version of the client does not support features which require mobile device's native properties. Once a native application is built there are two features at the top of our list.

The first one is using mobile device's camera to easily store a photo into HIP. This can probably be an extension of attachments and image widgets. There were ideas about adding a camera button into those widgets and clicking that button would initiate device's camera. Once a photo has been shot it would be stored either as attachment or uploaded as image depending on which widget this feature would be used on.

The second native feature will be notifications. When something relevant takes place in the workspace the asynchronous what's new list is not an immediate way to reach other users. Notifications would change this. Humap just has to be very careful not to send notifications about irrelevant actions happening in workspace. This will require both good default values for notifications and comprehensive options for users to define how they want to receive notifications. Once a user clicks a notification it will open HIP mobile client into corresponding document view (see Figure 18).

5 Afterword

In the first chapter in this thesis I claimed that I would show you how it is possible to create mobile a client that works in just about every mobile device, will offer the right features for mobile users and still you do not have to invest a huge amount of resources in it.

Reflecting back to those goals I would claim that our solution does indeed work in just about every mobile device. Of course, there are some old devices which are ruled out from supported devices, however, jQuery Mobile also supports a wide variety of even some older platforms.

Time will tell whether the features in the initial version of the mobile client are right or not. At least I feel like it is a good setup to start with. When designing the mobile client features we tried to take various factors into account. We had our research results including log file analysis and feedback survey, but also our own expertise and several years of background in developing Humap Tool and HIP. In the end our own gut feeling had also a somewhat significant role when designing and choosing the features. We knew that leaving out the edit features would reduce the time to release the first version, and we often work very iteratively. This felt like a good setup for the first set of features. We will definitely not stop improving the client based on our own plans and feedback that we will get from this version.

Working with jQuery Mobile and PhoneGap has been great. For anyone even bit familiar with jQuery using jQuery Mobile should not be an issue. Creating mock-up applications can be done in minutes. The first working prototype of our client with limited set of features was built in roughly ten hours. In this version users could login only to intra.humap.com, read what's new list and search and browse documents from Humap Software workspace. This already covered most of our features for the first release version of the mobile client. We also built this version into native application using PhoneGap, but there were some relative URL and cross-domain issues which we did not want to fix at this point. Nevertheless, building the application itself was extremely easy and we have quite clear plans how to fix those issues when we want to go for native application.

The most time consuming phase in this whole process was definitely technology and feature researches. Once we knew the technologies that we are going to work with and we had a good idea what features we wanted in the mobile client it was all downhill from there on. This is a good guideline to keep in mind in any software project. Design, specifications and communication will often take a major part of any project whereas the implementation itself might be done sometimes on a very fast schedule.

References

Adobe Announces Agreement to Acquire Nitobi, Creator of PhoneGap. 2011. Adobe press release. Accessed on 29th of May 2013. http://www.adobe.com/aboutadobe/pressroom/pressreleases/201110/AdobeAcquiresNitobi.html.

Balmer, D. 2012. Jo HTML5 Mobile App Framework. Jo application framework's website. Accessed on 29th of May 2013. http://joapp.com/.

CSS3 Media Queries. 2012. W3C Recommendation 19 June 2012. Accessed on 29th of May 2013. http://www.w3.org/TR/css3-mediaqueries/.

Dojo Mobile. 2013. Dojo Mobile website by Dojo Foundation. Accessed on 29th of May 2013. http://dojotoolkit.org/features/mobile.

EmbedJS. 2013. EmbedJS website by uxebu Consulting. Accessed on 29th of May 2013. http://uxebu.github.com/embedjs/.

Firtman, M. 2012. Programming the Mobile Web. US: O'Reilly Media.

GUI widget. 2013. Wikipedia. Accessed on 29th of May 2013. http://en.wikipedia.org/wiki/GUI_widget.

Havimäki, S. 2012. Yrittäjäksi yliopistosta. Jyväskylä: Jyväskylän yliopisto, Agora Center.

HIP Platform - Ainutlaatuinen alusta innovaatioiden jalostamiseen. 2013. Humap Store website. Accessed on 29th of May 2013. http://store.humap.com/products/hip-platform/.

Humap – New Ways of Working. 2012. Humap's website. Accessed on 29th of May 2013. http://www.humap.com/en/new-ways-of-working/.

Humap Inspiration Platform – HIP. 2012. Humap's website. Accessed on 29th of May 2013. http://www.humap.com/en/hip/.

Jimenez & Olalde 2013. Lungo framework's website. Accessed on 29th of May 2013. http://lungo.tapquo.com/.

jQuery Mobile. 2013. jQuery Mobile's website. Accessed on 29th of May 2013. http://jquerymobile.com/.

Key ICT indicators for developed and developing countries and the world. 2013. ITU (International Telecommunication Union). Accessed on 29th of May 2013. http://www.itu.int/ITU-D/ict/statistics/at_glance/KeyTelecom.html.

Kuler. 2013. Tool created by Adobe to explore, create and share color themes. Accessed on 29th of May 2013. https://kuler.adobe.com/.

La, N. 2011. Responsive Design with CSS3 Media Queries. Accessed on 29th of May 2013. http://webdesignerwall.com/tutorials/responsive-design-with-css3-media-queries.

LeRoux, B. 2012. PhoneGap, Cordova, and what's in a name? Accessed on 29th of May 2013. http://phonegap.com/2012/03/19/phonegap-cordova-and-what%E2%80%99s-in-a-name/.

Moth, D. 2013. Seven potential downsides from using responsive design. Accessed on 29th of May 2013. http://econsultancy.com/fi/blog/61870-seven-potential-downsides-from-using-responsive-design.

Operating system. 2013. Wikipedia. Accessed on 13th of April 2013. http://en.wikipedia.org/wiki/Operating_system.

PhoneGap. 2013. PhoneGap's website by Adobe. Accessed on 29th of May 2013. http://phonegap.com/.

Qualitative research. 2013. Wikipedia. Accessed on 29th of May 2013. http://en.wikipedia.org/wiki/Qualitative_research.

Quantitative research. 2013. Wikipedia. Accessed on 29th of May 2013. http://en.wikipedia.org/wiki/Quantitative_research.

Ramon. J. 2011. Sencha Touch or jQuery Mobile? – Read This Before You Make a Decision. Accessed on 29th of May 2013. http://miamicoder.com/2011/sencha-touch-or-jquery-mobile-read-this-before-you-make-a-decision/.

Reasons Mobile Matters. 2011. Go Mobile (GoMo) initiative led by Google. Accessed on 29th of May 2013. http://www.howtogetmo.com/en/d/why-go-mo/#reasons-mobile-matters.

Reid, J. 2011. jQuery Mobile. US: O'Reilly Media.

Responsive web design. 2013. Wikipedia. Accessed on 29th of May 2013. http://en.wikipedia.org/wiki/Responsive_web_design.

RhoMobile Suite. 2013. RhoMobile's website by Motorola. Accessed on 29th of May 2013. http://www.motorola.com/Business/US-EN/Business+Product+and+Services/Software+and+Applications/RhoMobile+Suite.

Sarmiento, J. 2011. 11 Reasons Why Responsive Design Isn't That Cool! 2011. Web-DesignShock. Accessed on 29th of May 2013. http://www.webdesignshock.com/responsive-design-problems/.

Sencha Touch. 2013. Sencha Touch's website. Accessed on 29th of May 2013. http://www.sencha.com/products/touch/.

Social media. 2013. Wikipedia. Accessed on 29th of May 2013. http://en.wikipedia.org/wiki/Social_media.

ThemeRoller For jQuery Mobile. 2013. Tool for designing jQuery Mobile themes. Accessed on 29th of May 2013. http://jquerymobile.com/themeroller/.

Titanium Mobile Development Environment. 2013. Titanium platform's website by Appcelerator. Accessed on 29th of May 2013. http://www.appcelerator.com/platform/titanium-platform/.

What Is Lean? 2009. Article on Lean Enterprise Institute's website. Accessed on 29th of May 2013. http://www.lean.org/whatslean/.